# Tumoroid Image Classification for Personalised Cancer Treatment

Florian Hübler, Viggo Moro, Fredrik Nestaas,
Johan Lokna and Tobias Wegel

Supervised by
Dr. Ines Lüchtefeld and Prof. Dr. Ce Zhang

Data Science Lab, ETH Zürich

February 17, 2023

**Abstract**

We present an automated tumoroid viability classification pipeline, which can be used for personalised cancer treatment. It entails an entire data preprocessing pipeline, which can be used to process microscopic images of collections of droplets containing tumoroids. The pipeline extracts droplets, normalises their images and then extracts the foreground. The classification is performed by a pretrained CNN, fine-tuned on a dataset containing colon cancer tumoroids. For the training, the dataset is filtered to not contain outliers. The best test accuracy we could achieve is at roughly 72%, with an $F_1$-score of around 0.72. To benefit from this erroneous model, we also report a statistical analysis on how medication choices can be based on the model. The code to this report can be found at https://gitlab.ethz.ch/twegel/ds-lab-tumor-classification.

# Contents

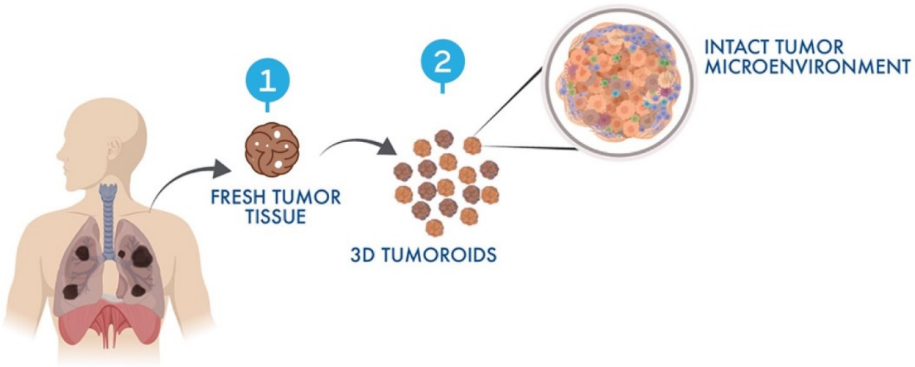


Figure 1: Tumoroid extraction: (1) Fresh tumor tissue is extracted from the patient which is then turned into (2) tumoroids with an intact tumor microenvironment, which can be used for ex-vivo drug trials. Figure from [Ehrhart, 2020].

# 1 Introduction

Modern oncological approaches try to tailor the treatment of cancer patients to the patient, rather than using a one-fits-all approach, in order to increase efficacy and/or reduce toxicity of the treatment, see for example [Jackson and Chester, 2014]. This is mostly motivated by inter-patient heterogeneity of tumor attributes, as well as intra-patient, inter-tumor heterogeneity and even intra-patient, intra-tumor heterogeneity. As part of such personalised cancer treatments, it can be of interest to choose a medication from a set of plausible drugs specific to the patient's tumor, rather than based on population estimates of the drug's performance. However, it is not obvious how to decide (with high confidence) that one drug might be more effective than another for treating a specific patient's tumor. One approach to tackling this problem is by attaining tissue from the tumor that is to be treated. From this tissue, multiple tumoroids, whose size is on the micrometer scale, can be extracted and grown in a culture to attain intact tumoroid microenvironments (mini tumors). These tumoroids represent the complex network of interactions, molecular and the structural heterogeneity of the tumor [Lê et al., 2022]. Figure 1 visualises the process of extracting tumoroids. Once the tumoroids have been extracted, each can be exposed to a different drug. If the number of cancer cells or tumoroids killed off by one of the drugs is significantly higher than for other drugs, it might be expected that the treatment of the patient with this drug could be more successful.

However, currently the tumoroid assays require 2-3 months of culture time, which makes them infeasible for fast clinical decision making. Therefore, new technologies to speed up this process are required. To this end, a research collaboration between the Inselspital Bern, the University of Bern, the Functional Immune Repertoire Analysis group as well as the Tumor and Stem Cell Dynamics group at ETH Zürich has developed a method to separate tumoroids into small drug-loaded nanoliter-droplets of a carrier fluid, where the drug can be varied per droplet. The aim is to dramatically accelerate the culture process and optimise the prediction of drug susceptibility.

There are multiple technical problems that arise in this procedure, among which is the classification of the viability of the cells present in a tumoroid after treatment with a

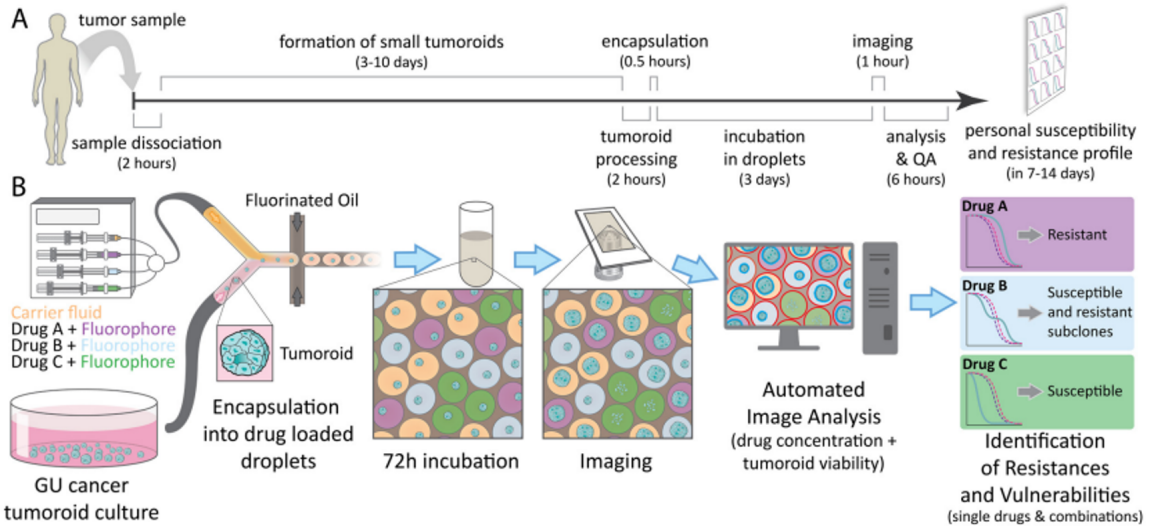


Figure 2: The entire drug testing pipline. First, the tumoroid culture is encapsulated into drug-loaded droplets that also contain a fluorophore that marks which drug is in which droplet. The droplets are separated using a fluorinated oil and then incubated, before imaging them using a microscope. Afterwards, the automated image analysis phase takes place, which outputs a statistical analysis of the different drug effects. This last phase is described in this report.

drug. Currently, the droplets are loaded with a fluorophore unique to each drug in order to identify the drug per droplet. These droplets are then incubated for a certain time-period (e.g., 72 hours) and afterwards images of the collection of droplets are attained via a microscope. So far, the evaluation of such images has been done manually by adding another solution to the droplet which acts as a visual indicator for whether a cell is dead or alive. Therefore, the images so far contained an additional colour-channel, of which the intensity was used to classify the tumoroids' viability by hand. The goal of this sub-project is to circumvent this rather time-costly and tedious work by classifying the microscopic images of the droplets through automated image analysis, using computer vision systems. Figure 2 visualises the entire pipeline and shows where the automated image analysis comes into play.

A key insight for using an automated image analysis for this purpose is, that the viability per tumoroid is not the end-goal: The decision on which drug to use for the source tumor is. Therefore, we provide a statistical analysis on top of the viability classification, based on which the decision which drug is preferable can be made. Crucially, this statistical analysis includes the error induced by the classification model, so that the use of viability classification for deciding which drug to use becomes more meaningful and has a notion of confidence in the decision.

**Structure.** The rest of the report is structured as follows. We first give an in-depth description of the problem and dataset in Section 1.1. Section 2 contains the methods

Figure 3: Raw data input from the microscope.

used for viability classification. In Section 3 we describe the statistical analysis that can be performed on top of the classification model. Section 4 presents the results from classification model as well as statistical analyis on our dataset. Finally, in Section 5 we discuss the outcome of our project and assess its quality.

## 1.1 Problem Description

The images received from the microscope include a view that contains multiple droplets of varying size. An example of an image received from the microscope can be seen in Figure 3.

The automated image analysis has to solve three problems. Firstly, it has to identify the droplets within the image and return their location, that is, return two coordinates that give the centre pixels of the droplet in the image. This task was already mostly solved when starting this project, so we will not go into detail here. Secondly, the image analysis has to decide which part of the image around this centre is relevant to the classification problem, which we assume to be just the droplet and its content, and extract this part from the image. Finally, it has to assign a label to each droplet. We experimented with different types of labels, but each were combinations of the four presented below:

$0$ - The droplet is overcrowded or empty,
$1$ - all cells in the droplet are dead,
$2$ - all cells in the droplet are alive,
$3$ - some of the cells are alive, some are dead.

**Dataset.** To solve the latter two problems, we were provided a dataset containing 42 images of multiple droplets with colon cancer tumoroids. An example can be seen in

Figure 3. The microscope takes image-blocks of pixel-size $1015 \times 1015$. In the standard acquisition process, the microscope then takes $6 \times 3$ of these blocks and stitches them together into a $6090 \times 3045$ image. The borders between these blocks can sometimes be fairly visible, see Figure 3. However, occasionally, the acquisition was done differently in that only subregions of the droplet container were imaged, so that the image sizes sometimes deviate. This was done due to focusing issues: If the droplet container was not flat enough under the microscope, the focus depth would shift along certain axes in the image, as can also be seen in Figure 3 from left to right. Whenever this effect was too strong, the alternate acquisition was performed. Therefore, each image contained a varying number of droplets roughly between 100 and 400, and on average 270 droplets, which gave us a total number of 11.340 droplets. Since the localisation of droplets within the image was already solved, the dataset also contained (sometimes faulty) positions of droplets within the image.

Furthermore, the dataset also included labels for these droplets, which were hand-labelled into the four classes mentioned above using an indicator solution which flags dead cells, as described before. We were therefore able to use supervised learning techniques. Throughout the dataset, there are quite strong class imbalances, with different imbalances between different images. Overall, there were 3313 droplets of class 0: overcrowded or empty, 5592 droplets of class 1: all dead, 1474 droplets of class 2: all alive and 828 droplets of class 3: some cells are alive, some are dead, see Figure 4.

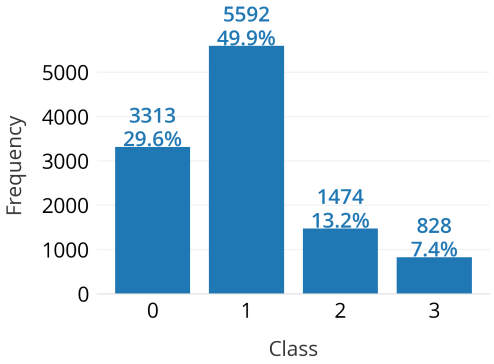


Figure 4: Class imbalance of the dataset.

**Classification Problem.** We assume throughout this report that the way dead or alive tumoroids look is independent of the drug to which they were exposed. This means that we can treat the classification problem of classifying droplets into one of the four classes and doing a statistical analysis to determine the best drug separately. For training, once a training dataset $\{(x_i, y_i)\}_{i=1}^n$ of droplet images $x_i$ and labels $y_i \in \{0, 1, 2, 3\}$ is created from the provided dataset, we have to learn an estimator $f$ that predicts $y \in \{0, 1, 2, 3\}$ for a new, unseen input $x$.

**Decision Problem.** Once we have obtained such a model $f$, since any model is prone to be erroneous, part of the problem is to estimate how well each drug performs taking into account the statistical error by the model $f$ as well as noise in the data. In particular, given a set of drugs $\{1, ..., d\}$, a dataset of droplets $\{x_i\}_{i=1}^m$ and a trained, fixed classification model $f$ which classifies each droplet as $\hat{y}_i \in \{0, 1, 2, 3\}$, we have to estimate the best drug from the set of drugs. "Best" will be measured by the population probability of killing cells in a tumoroid, given a drug, which in turn we must estimate.

# 2 Methods

The process with which we obtained our model and the preprocessing pipline can roughly be broken down into the following six steps. 1. Image preprocessing, 2. Extraction of Droplets, 3. Droplet preprocessing, 4. Modelling, 5. Training and 6. Evaluation. In the following sections, we will describe each step in detail.

## 2.1 Image Preprocessing

The data was gathered using a Nikon microscope and hence exists in its natural *.nd2* file-format. The first step was to transform these into machine learning compatible formats (pytorch tensors). Afterwards the image intensities were normalized using the standard technique called Nyúl's method [Nyúl and Udupa, 1999], which matches quantiles of the intensity histogram to a fixed target histogram. Intensities between these quantiles are linearly interpolated. Here the target histogram is the average histogram of the training set. Furthermore the intensities were linearly mapped into the interval $[0, 1]$. The issues of bordering image patches as well as different focussing depths were not tackled on the image level, but rather on the droplet level. Figure 5 shows an example of an input image and its normalised version.

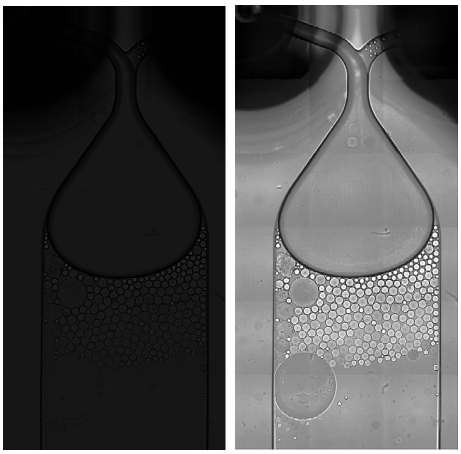

Figure 5: Sample input image and the normalised version of the image. Note that the input image is so dark, because a small number of pixels have a vastly higher intensity compared to the rest.

## 2.2 Extraction of Droplets

For the extraction of the droplet from the images, we used a custom algorithm. The algorithm is based on an edge search, starting from the centre of the droplet. From this centre, a number of rays are spaced uniformly distanced over all angles around the centre and pointing outwards. Along these rays, an edge search is done by discretizing the ray and calculating the intensity gradient on each ray between all neighbouring pairs of discretization steps. If the intensity gradient is very high at any point, it will be flagged as the edge of the droplet. Therefore, we get a set of points that lie on the edge of the droplet. To not include any faulty points, we only kept the points whose distances are within the 5% and 95% quantiles - numbers we found most effective. Afterwards, the convex hull of the remaining points is kept, the rest of the image is discharged. The results of this algorithm are very promising, given its complexity. Almost always the droplet is exactly extracted up to small fragments. An example of a droplet that was extracted using this algorithm can be seen in Figure 6.

The main failure modes of the extraction algorithm occur, whenever the centre of the droplet is wrongly determined beforehand, or if there is some artefact (like a satellite droplet) obscuring the view on the droplet. This is the case, because then the edge of

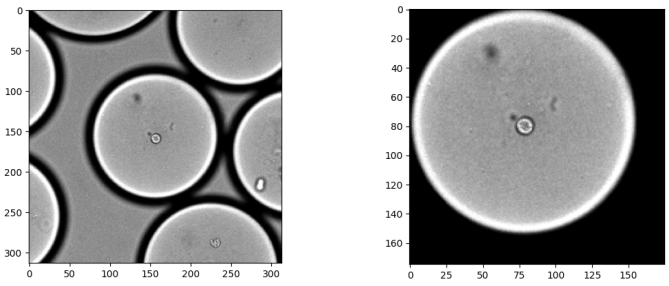

Figure 6: Droplet before and after the extraction.

the droplet might have a different gradient than expected, or because the high gradient occurs too early or to late. In Section 2.3 we describe, how these failures of the droplet extraction are handled.

## 2.3 Droplet Preprocessing

After the extraction process has finished, final preprocessing steps are performed on the droplet level. Firstly the *bias field*[1] is removed, before using padding to create unit sized images. Finally we perform outlier detection to remove *bad* droplets.

**Bias Field Removal.** Although great efforts were put into place to guarantee uniform image properties throughout the whole image, different focusing depths still produce slight intensity shifts in each block. This phenomenon can for example be observed in Figure 3 when looking at the borders of the $6$ slices that make up the whole image. When the effect became too severe, images were manually split up in parts during the acquisition process. One of these input images can be seen in Figure 7. In both cases the change in intensities can be modelled by an multiplicative bias field. Mathematically, an observed intensity at coordinates $x$ is given by

$$j(x) = i(x) \cdot b(x),$$

where $i(x)$ is the true underlying pixel intensity and $b(x)$ the multiplicative bias field at $x$. One typically assumes that this bias field $b$ is smooth, which is empirically supported by our images. In order to restore the actual intensities $i(x)$, we use the the special structure of our problem. Since a single droplet is tiny compared to the whole image, we use the smoothness of $b$ and model it as constant ($b(x) \equiv b$) on a single droplet. We furthermore use the fact that we normalised the intensities into $j(x) \in [0, 1]$ beforehand. We therefore can estimate our $b$ over a droplet $D$ by

$$b = \max_{x \in D} j(x).$$

[1]The concept of bias fields is typically used in magnetic resonance imaging (MRI). For an overview we refer to [Vovk et al., 2007].

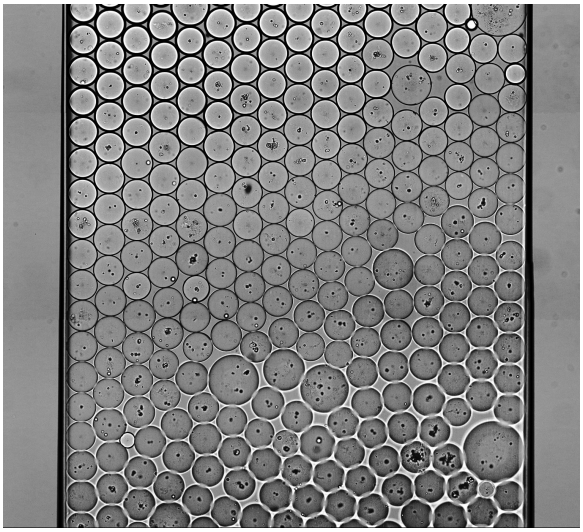

Figure 7: Example of an preprocessed input image, which was manually split by the microscope operator due to its heavy focus shift.

In other words, the bias field restricts our droplets' intensities into the interval $[0, {}^{b}/_{B}]$ where $B$ is the maximum value the bias field takes on the whole image. A simple min-max intensity normalisation to $[0, 1]$ hence returns the true intensities up to a factor that is constant over all droplets, effectively removing the bias field.

**Padding.** Since many machine learning models require a fixed input image size, we padded the single droplets to a unit size. This was achieved by either planting the droplets in the top left corner of a fixed $170 \times 170$ black canvas, or centred droplets with uniform padding in all directions. Droplets that were bigger than the predefined size were ignored.

**Outlier detection.** The goal of outlier detection is to remove any samples from the training data set that are statistically so different, that we assume they will hurt the generalisation performance of our model, rather than improve it. We used a combination of two outlier detection methods, namely Isolation Forest as well as Local Isolation Factor, and flagged any sample as an outlier that was detected by either method. Overall, our approach was to be rather sensitive to outliers - meaning that if a "droplet" was faulty, we most likely flagged it - with the downside of a low specificity - meaning that we flagged a lot of droplets as outliers which might have been okay to use. The reasoning behind this high-sensitivity, low-specificity trade-off is that 1) our dataset of roughly 11.300 droplets was quite large and that 2) the signal-to-noise ratio regarding vi-

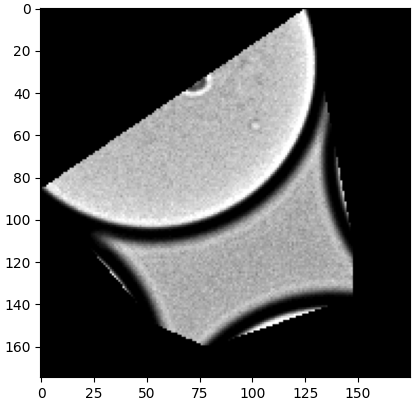


Figure 8: Example "droplet" that was flagged by the outlier detection.

ability seemed to be rather bad already. Overall, we removed roughly 30% of the data using this outlier detection method. A manual inspection (see Figure 8) showed that this was particularly effective at removing samples where one of the previous steps, like the foreground extraction, failed.

**To summarise,** Sections 2.1 to 2.3 describe what we call the preprocessing pipeline. Three sample droplets that are created using this pipeline can be seen in Figure 9.

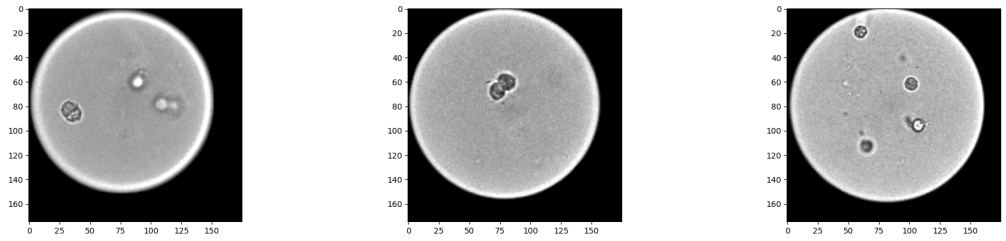

Figure 9: Three sample droplet images after the complete preprocessing pipeline.

## 2.4 Modelling

Under modelling, we understand everything that concerns the choice of model, and its hyperparameters.

We experimented with different pretrained convolutional neural network CNN backbones from the pytorch libraries `timm` and `torchvision` as well as some pretrained visual transformers with an additional head to solve the classification problem. In the end, we used the Resnet34 by [He et al., 2015], implemented in the `timm` library, having experimented with different architectures and hyperparameters. Note that using the pretrained version allowed us to significantly reduce the training time, as the initial parameters in the neural network have been trained to work well on other tasks already (see [He et al., 2015]).

## 2.5 Training

Next to the pre-processing described in Section 2.1 and Section 2.3, we also apply training time augmentations to the data. This means that we randomly flip, rotate and add color jitter to the images before passing them to the model, aiming to simulate having a larger data set available.

The objective of the optimisation was to minimize the the sample risk over the model parameters $\theta$, defined via the weighted cross-entropy loss:

$$-\frac{1}{n\sum_{j=1}^{C} w_j}\sum_{i=1}^{n}\sum_{j=1}^{C} w_j \mathbb{1}\left(y_i = j\right)\log\left(\hat{y}_j\right)$$

Here $y_i$ denotes the label, which was in $\{1,\ldots,C\}$, i.e. either in $\{1,2,3,4\}$ or $\{1,2\}$, depending on whether we trained a 1-vs-all model or a $4$-class model. $n$ denotes the

number of samples, which for the training dataset was at roughly $7800$, and $w_y \geq 0$ denotes a weight per class, putting more emphasis on the under-represented classes. The variables $y_i$ and $\hat{y}_i$ denote the true and predicted class of the $i$-th sample, respectively. In order to obtain $\hat{y}_i$, we use the softmax function on the neural network outputs. This is a differentiable approximation of the maximum function, and is defined as

$$\text{Softmax}\left(x_j\right) = \frac{\exp\left(x_j\right)}{\sum_{k=1}^{C} \exp\left(x_k\right)},$$

where $x \in \mathbb{R}^C$ is an output of the neural network.[2]

In order to optimize the objective, we used the Adam optimizer [Kingma and Ba, 2014] with batch size $50$. We tuned the model for 10 epochs using a learning rate of $10^{-4}$, not freezing any parameters, i.e. letting any parameters update according to the optimization scheme.

## 2.6 Evaluation

To evaluate our models, we used several metrics; the test sample risk via weighted cross-entropy (see training objective), the accuracy of our model, and in order to compensate for the class imbalance described in Section 1.1 a macro-average of the class-internal $F_1$-scores, which is defined as

$$F_{\text{macro}} = \frac{1}{C} \sum_{y=1}^{C} 2 \cdot \frac{\text{Precision}_y \cdot \text{Recall}_y}{\text{Precision}_y + \text{Recall}_y}.$$

Here $C$ again denotes the number of classes (which was either $2$ or $4$) and now $n$ is the sample size of the test set, $\text{Precision}_y$ and $\text{Recall}_y$ are the normal definitions of precision and recall, for each class $y$.

[2]Note that the regular max function is defined as $\text{Max}\left(x_i\right) = x_i \mathbb{1}\left(x_i \geq x_j \forall j\right)$

# 3 Statistical Analysis

As before, we are assuming that all samples are from the same tumor of the same patient.

## 3.1 Population Model

Consider the label-assigning, deterministic function $Y$ that maps from the image space $\mathcal{X}$ to the labels $\{1, \dots, C\}$. The map $Y : \mathcal{X} \to \{1, \dots, C\}$ therefore maps images $x \in \mathcal{X}$ of droplets to the *true* label $Y(x)$. Say we have a set of $m$ predetermined drugs that we want to test during inference time, and that are denoted by $d \in \{1, ..., m\}$. For each drug $d$, define $P^d$ to be a (unknown) probability distribution on $\mathcal{X}$ and define the population random image reveived from this drug $d$ as well as the corresponding label as

$$X^d \sim P^d, \qquad Y(X^d) \sim \pi^d,$$

where $\pi^d$ is a distribution on $\{1, \dots, C\}$ induced by $P^d$. Our goal is to estimate $\pi^d$ for all drugs $d$, because then we can determine the best drug by choosing the one with the highest probability for the label "all dead". That is, the drug which has the highest probability of killing all tumoroids in a droplet. Further, denote the prediction model $f$ that predicts labels for the images, that is, $f$ is also a map $f : \mathcal{X} \to \{1, \dots, C\}$. $f$ is fixed and is independent of all random variables we consider. Denote the distribution of $f(X^d)$ as

$$f(X^d) \sim \nu^d$$

such that $\nu^d$ is a probability distribution on $\{1, \dots, C\}$.

Additionally, we denote for any drug $d'$ (no matter if one of the drugs above) the distribution of $Y(X^{d'})$ given that $X^{d'}$ was predicted to be $f(X^{d'}) = \ell$ as

$$Y(X^{d'}) | f(X^{d'}) = \ell \sim \mu^{d',\ell} \equiv \mu^{\ell}.$$

Above, we assume that $\mu^{d',\ell}$ is invariant under different drugs $d'$. This makes sense, because we do not expect dead or alive tumoroids to look different depending on which drug they were exposed to. Therefore, we we assert $\mu^{d',\ell} \equiv \mu^{\ell}$.

## 3.2 Estimation

We have to split our estimation task into two parts: In the first part we have to quantify the error of the model $f$ by estimating $\mu^{\ell}$. We therefore need a test set for which we also have access to the labels. In the second estimation part, we estimate $\nu^d$ and then decide between new drugs (not necessarily the ones with which we estimated $\mu^{\ell}$) by using the estimated $\mu^{\ell}$ and $\nu^d$ to estimate $\pi^d$. In this part we only need access to a test set without labels associated with the images. In practice, the first part is part of the "training", whereas the second part is part of "inference".

**Part 1: Quantifying model error.** In this first part, we can neglect the dependence on the drug, because we only estimate $\mu^\ell$. Say we have an i.i.d. dataset $\mathcal{D} = \{X_1, ..., X_n\}$ of samples from some distribution $P$, which for example could be a mixture of distributions $P^{d'}$ and some collection of drugs $d'$. Furthermore, assume we have access to the labels $Y(X_i)$ for all $i$. Define

$$\mathcal{D}_\ell = \{X \in \mathcal{D} \mid f(X) = \ell\}.$$

Then we can define the unbiased estimator $\hat{\mu}^\ell$ as

$$\begin{aligned}\hat{\mu}^\ell(\{k\}) :&= \frac{|\{X \in \mathcal{D}_\ell \mid Y(X) = k\}|}{|\mathcal{D}_\ell|} \\ &= \frac{\#\text{ droplets labeled as } k \text{ and predicted as } \ell}{\#\text{ droplets predicted as } \ell}.\end{aligned}$$

It is easily seen that this estimator is unbiased by noting that

$$\mathbb{E}\hat{\mu}^\ell(\{k\}) = \frac{1}{|\mathcal{D}_\ell|}\sum_{X\in\mathcal{D}_\ell}\mathbb{E}\mathbb{1}\left(Y(X) = k\right) = \frac{|\mathcal{D}_\ell|}{|\mathcal{D}_\ell|}\mathbb{P}\left(Y(X) = k \mid f(X) = \ell\right) = \mu^\ell(\{k\}).$$

**Part 2: Estimating drug effect.** Assume that for the collection of drugs $d \in \{1, \dots, m\}$, we have i.i.d. datasets $\mathcal{D}^d = \left\{X_1^d, ..., X_{n_d}^d\right\}$, which were sampled from $P^d$. Crucially, these are datasets from inference time, where we want to decide which drug to use and they can be different datasets from part one. We can estimate $\nu^d$ with the unbiased estimator $\hat{\nu}^d$

$$\begin{aligned}\hat{\nu}^d(\{k\}) &:= \frac{\left|\left\{X \in \mathcal{D}^d \;\middle|\; f(X) = k\right\}\right|}{|\mathcal{D}^d|} \\ &= \frac{\#\text{ droplets exposed to drug } d \text{ and predicted class } k}{\#\text{ droplets exposed to drug } d}.\end{aligned}$$

Again, it is quite obvious that it is unbiased:

$$\mathbb{E}\hat{\nu}^d(\{k\}) = \frac{1}{|\mathcal{D}^d|}\sum_{X\in\mathcal{D}^d}\mathbb{E}\mathbb{1}\left(f(X) = k\right) = \frac{|\mathcal{D}^d|}{|\mathcal{D}^d|}\mathbb{P}\left(f(X^d) = k\right) = \nu^d(\{k\}).$$

To derive the estimator for $\pi^d$, we first have to realise that by the law of total probability

$$\begin{aligned}\pi^d(\{k\}) &= \mathbb{P}\left(Y(X^d) = k\right) \\ &= \sum_{\ell=1}^{C}\mathbb{P}\left(Y(X^d) = k \;\middle|\; f(X^d) = \ell\right)\mathbb{P}\left(f(X^d) = \ell\right) \\ &= \sum_{\ell=1}^{C}\mu^\ell(\{k\})\nu^d(\{\ell\}).\end{aligned}$$

Therefore, with slight abuse of notation, we have

$$\pi^d = \begin{pmatrix}\pi^d(\{1\}) \\ \vdots \\ \pi^d(\{C\})\end{pmatrix} = \begin{pmatrix}\mu^1(\{1\}) & \dots & \mu^C(\{1\}) \\ \vdots & \ddots & \vdots \\ \mu^1(\{C\}) & \dots & \mu^C(\{C\})\end{pmatrix}\begin{pmatrix}\nu^d(\{1\}) \\ \vdots \\ \nu^d(\{C\})\end{pmatrix} = \mu\nu^d.$$

Note that $\mu$ is what is commonly referred to as the true confusion matrix of $f$. It is now quite natural to define the estimator

$$\hat{\pi}^d = \hat{\mu}\hat{\nu}^d$$

where we again slightly abuse notation to define

$$\hat{\mu} = \begin{pmatrix} \hat{\mu}^1(\{1\}) & \dots & \hat{\mu}^C(\{1\}) \\ \vdots & \ddots & \vdots \\ \hat{\mu}^1(\{C\}) & \dots & \hat{\mu}^C(\{C\}) \end{pmatrix} \quad \text{and} \quad \hat{\nu}^d = \begin{pmatrix} \hat{\nu}^d(\{1\}) \\ \vdots \\ \hat{\nu}^d(\{C\}) \end{pmatrix}.$$

$\hat{\mu}$ is now the estimated confusion matrix. Note that also $\hat{\pi}^d$ is unbiased, since by the independence of $\hat{\mu}^\ell$ and $\hat{\nu}^d$ we have $\mathbb{E}\hat{\pi}^d = \mathbb{E}\hat{\mu}\hat{\nu}^d = \mathbb{E}\hat{\mu}\mathbb{E}\hat{\nu}^d = \mu\nu^d$.

## 3.3 Confidence Bounds

To understand our estimators better, we also report confidence intervals. Crucially, we will use these confidence intervals to determine that a drug is better than another *with high probability*. The derivation of the following confidence intervals can be found in Appendix A. Define the minimal sub-dataset size

$$s_d = \min_\ell \left\{ |D_\ell|, \left|D^d\right| \right\},$$

and for $\alpha \in [0,1]$, define the deviation term

$$c_{d,k}(\alpha) = \sqrt{\frac{(C+1)^2 \log(4C/\alpha)}{2s_d}}.$$

If $C = 2$, let

$$c_{d,k}(\alpha) = \sqrt{\frac{2\log(6/\alpha)}{s_d}}.$$

If you assert that $\hat{\mu} \equiv \mu$, that is, there is no estimation error of the confusion matrix, let

$$c_{d,k}(\alpha) = \sqrt{\frac{\log(6/\alpha)}{2s_d}}.$$

Further, define the interval

$$C_{d,k}(\alpha) = \left[\max\left\{\hat{\pi}^d(\{k\}) - c_{d,k}(\alpha), 0\right\}, \min\left\{\hat{\pi}^d(\{k\}) + c_{d,k}(\alpha), 1\right\}\right].$$

Then it holds that

$$\mathbb{P}\left(\pi^d(\{k\}) \in C_{d,k}(\alpha)\right) \geq 1 - \alpha.$$

Let $C_{d,k}(\alpha/m)$ be the confidence intervals from above, now for confidence level $1 - \frac{\alpha}{m}$. Then analogous to Bonferroni's principle, we have that

$$\mathbb{P}\left(\pi^d(\{k\}) \in C_{d,k}(\alpha/m)\ \forall d = 1, ..., m\right) \geq 1 - \alpha.$$

See Appendix A for a proof. Therefore, the true parameters falling into the confidence intervals uniformly for all $d$ happens with probability at least $1 - \alpha$. In Appendix B we discuss how one can use the confidence intervals to determine necessary sample size.

## 3.4 Sampling Algorithm

In order to actually estimate drug effects and to be able to say with high confidence which of the $m$ drugs is the most effective, with probability $1-\alpha$, one could follow the approach below:

1. Fix a confidence level $\alpha$.
2. Sample droplets until the $1-\frac{\alpha}{m}$ confidence interval $C_{d,k}(\alpha/m)$ of the drug $d$ with the highest estimate $\pi^d(\{k\})$ does not overlap any other confidence intervals.
3. Then $\pi^d(\{k\}) > \pi^{d'}(\{k\})$ for all other drugs $d'$ with probability at least $1-\alpha$.

Depending on the assumptions made and the effort that it takes to sample more droplets, other sampling algorithms might be favourable. This approach is probably the most conservative one.

# 4 Results

## 4.1 Classification Performance

We tested the performance of our final classification CNN on a held-out test subset from the colon cancer dataset. We report the results for the best 4-class model and the best binary (class 1 vs. rest) model. The results can be seen in the form of a confusion matrix in Figure 10.

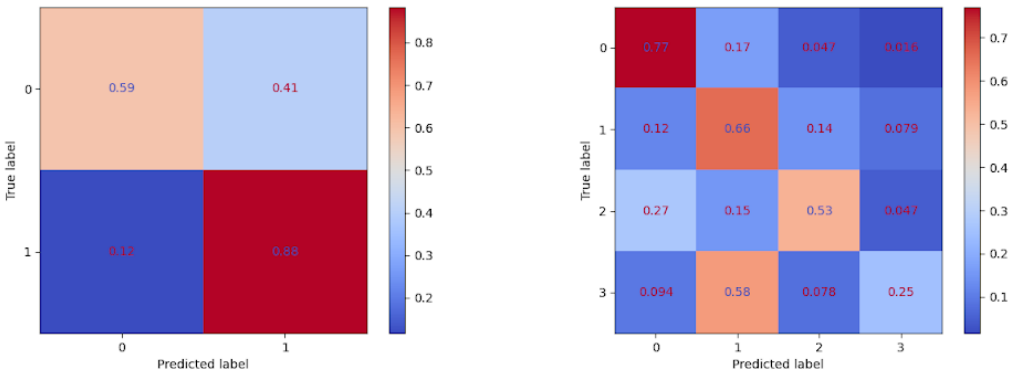


Figure 10: Confusion matrices for the best binary and multi-class classification models.

One can clearly see the class imbalance having a big impact on the model's classification performance. Furthermore, class 3: partially dead seems to be confused or not predicted at all, which conceptually also seems plausible. This was also the motivation to try a 1-vs-all approach. The metrics described in Section 2.6 for the two models are reported in Table 1. One can clearly see that both models do not perform too well, which we attribute at least partially to the problem setting itself.

| | Binary | 4-class |
|---|---|---|
| $F_{\text{macro}}$ | 0.7201 | 0.5100 |
| accuracy | 0.7290 | 0.6403 |
| weighted cross-entropy | 0.5743 | 1.1582 |

Table 1: Results for best 1-vs-all model (class 1) and best 4-class model. Weights for the cross-entropy are $\frac{10}{3}, 2, \frac{20}{3}, \frac{20}{3}$ for class 0, 1, 2, and 3 respectively when performing multi-class classification and $1$ for both classes when performing binary classification. We chose the weights equal to the reciprocal of the fraction of the dataset constituted by each class; in the case of binary classification, the classes are more or less balanced.

### 4.2 Statistical Analysis

Figure 11 shows an example of the estimated drug effect for an example nd2 image estimated at level $\alpha = 0.2$, following steps 1 and 2 in Section 3.4. From this data, it is not possible to determine which drug is the best one with probability $1 - \alpha = 0.8$, because the estimated effect of drug 0 overlaps with that of drug 1. However, comparing the leftmost and rightmost figures, knowing the confusion matrix perfectly allows for significant gains in terms of estimation uncertainty.[3]

Indeed, we were not able to find an example in the data set where it is possible to say which drug is the best. In our view, the best way to improve the confidence intervals would be to gather more data for each drug, so that the effect of each can be estimated more precisely. Appendix A gives the mathematical reasons why and quantifies to what degree additional data will help. For an overview of the required sample sizes using different estimation strategies, please refer to Figure 12 in the Appendix.

## 5 Discussion

We present two main contributions in this report. First, we describe the design and training of a machine learning model to classify mini-tumor droplets, in order to effectivize personalized cancer treatment assignment. Second, we provide a statistical analysis of the errors the model makes, in order to estimate the efficacy of each drug. We acknowledge that the approach has some limitations, which we discuss in the following.

A first factor to consider is the class imbalance in the data set, where the most common class is present nearly half of the time, while the least common class is present only 7.5% of the time. If a model then optimizes for being correct as often as possible, it will tend to predict the majority class over the rarer ones. There are several approaches to tackling this issue, and in this report, we explored using a weighted cross entropy loss. While this does have a negative impact on the accuracy of the classifier, it favors predicting less common classes, which we believe is favorable to a case where they are very rarely or even never predicted.

Further, in Section 4.2, when estimating the drug effect on real data, we see that the

[3] As can be seen in Appendix A, we apply the union bound to some of the probabilities. This is often a "pessimistic" bound, resulting in the present high uncertainty.

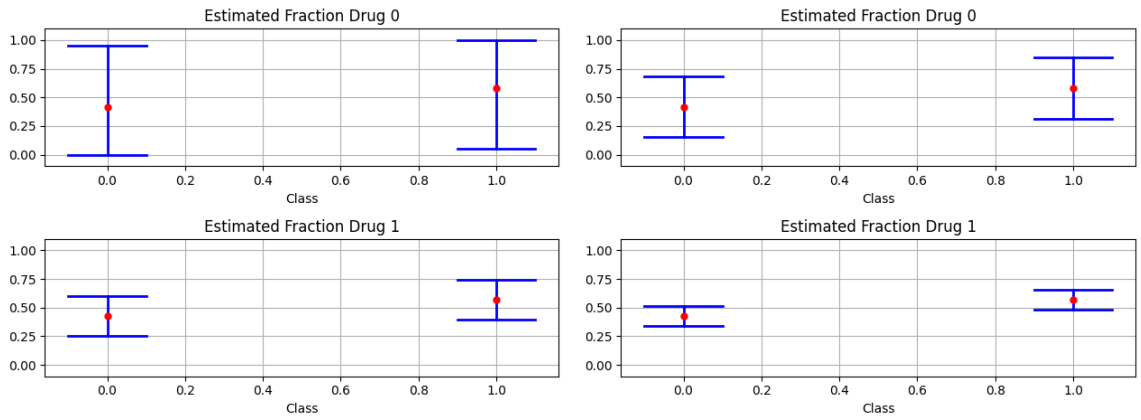


Figure 11: Example confidence intervals from a data set with two drugs present. The two leftmost of the figures show the confidence intervals obtained when accounting for the uncertainty in estimating the confusion matrix. On the other hand, the rightmost figures assume that the confusion matrix is known. Note that we compare two classes, in which class 0 denotes the case that not all of the cells are dead, while class 1 denotes the case that all of the cells are dead. The red dot denotes the point estimate of the probability of a certain class appearing, while the blue lines give the $1 - \alpha = 0.8$ confidence sets.

amount of available data limits the confidence in the estimates. Gathering more data to perform these estimations is, in our view, the best way to deal with this problem, as it will allow for more accurate estimation of each drug's effect. We do remark, however, that the provided confidence intervals are pessimistic in nature, as they are limited for each drug to estimate by the smallest number of samples available for each class. That being said, they do quantify an upper bound on the uncertainty under the assumptions made in Section 3.

Finally, we observed a number of anomailies in the data itself, and ended up removing around 30% of the available training data. While the outlier and anomaly detection methods introduced seem to do well in filtering out bad training samples, a cleaner data set will generally benefit the performance of machine learning models. As such, we think that any model's predictive performance probably will be at most decent, because noise in the data and bad labelling have made the classification task only solvable to a certain degree. However, we believe that using the statistical analysis presented in Section 3 can make the enduser gain a lot more value from the model.

# References

[Ehrhart, 2020] Ehrhart, J. (2020). Sting: rational drug combinations development. `https://eolas-bio.co.jp/lit/nilogen/Application-Note-May-2021-STING.pdf`. Accessed: 2022-12-19.

[He et al., 2015] He, K., Zhang, X., Ren, S., and Sun, J. (2015). Deep residual learning for image recognition.

[Jackson and Chester, 2014] Jackson, S. E. and Chester, J. D. (2014). Personalised cancer medicine. *Int J Cancer*, 137(2):262–266.

[Kingma and Ba, 2014] Kingma, D. P. and Ba, J. (2014). Adam: A method for stochastic optimization.

[Lê et al., 2022] Lê, H., Seitlinger, J., Lindner, V., Olland, A., Falcoz, P.-E., Benkirane-Jessel, N., and Quéméneur, E. (2022). Patient-Derived lung Tumoroids-An emerging technology in drug development and precision medicine. *Biomedicines*, 10(7).

[Nyúl and Udupa, 1999] Nyúl, L. G. and Udupa, J. K. (1999). On standardizing the mr image intensity scale. *Magnetic Resonance in Medicine*, 42(6):1072–1081.

[Vovk et al., 2007] Vovk, U., Pernus, F., and Likar, B. (2007). A review of methods for correction of intensity inhomogeneity in MRI. *IEEE Trans Med Imaging*, 26(3):405–421.

[Wainwright, 2019] Wainwright, M. J. (2019). *High-Dimensional Statistics: A Non-Asymptotic Viewpoint*. Cambridge Series in Statistical and Probabilistic Mathematics. Cambridge University Press.

# Appendices

## A Proofs

**Lemma 1** (Confidence Intervals.)**.** *Define the minimal sub-dataset size*

$$s_d = \min\left\{\min_{\ell}\left\{\left|D_\ell\right|\right\}, \left|D^d\right|\right\},$$

*and for* $\alpha \in [0,1]$*, define the deviation term*

$$c_{d,k}(\alpha) = \sqrt{\frac{(C+1)^2 \log\left(4C/\alpha\right)}{2s_d}}.$$

*Further, define the interval*

$$C_{d,k}(\alpha) = \left[\max\left\{\hat{\pi}^d(\{k\}) - c_{d,k}(\alpha), 0\right\}, \min\left\{\hat{\pi}^d(\{k\}) + c_{d,k}(\alpha), 1\right\}\right].$$

*Then it holds that*

$$\mathbb{P}\left(\pi^d(\{k\}) \in C_{d,k}(\alpha)\right) \geq 1 - \alpha.$$

*Proof.* First we bound the deviation of $\hat{\mu}^\ell$ and $\hat{\nu}^d$ for some specific label $k$, using Hoeffdings inequality (See Proposition 2.5, [Wainwright, 2019]). Through simple plug-in, we get that

$$\begin{aligned}
\mathbb{P}\left(\left|\hat{\mu}^\ell(\{k\}) - \mu^\ell(\{k\})\right| \geq t\right) &\leq 2\exp\left(-2t^2\left|D_\ell\right|\right) \\
\mathbb{P}\left(\left|\hat{\nu}^d(\{k\}) - \nu^d(\{k\})\right| \geq t\right) &\leq 2\exp\left(-2t^2\left|D^d\right|\right).
\end{aligned}$$

Through union bound arguments, we get that

$$\begin{aligned}
\mathbb{P}\left(\exists \ell \in \{1,\ldots,C\} : \left|\hat{\mu}^\ell(\{k\}) - \mu^\ell(\{k\})\right| \geq t\right) &\leq 2C\exp\left(-2t^2\left|D_\ell\right|\right) \\
\mathbb{P}\left(\exists k \in \{1,\ldots,C\} : \left|\hat{\nu}^d(\{k\}) - \nu^d(\{k\})\right| \geq t\right) &\leq 2C\exp\left(-2t^2\left|D^d\right|\right).
\end{aligned}$$

We now have to bound the deviation of $\hat{\pi}^d$. Recall the definition

$$\hat{\pi}^d(\{k\}) = \sum_{\ell=1}^{C} \hat{\mu}^\ell(\{k\})\hat{\nu}^d(\{\ell\}).$$

Define, with slight abuse of notation,

$$\hat{\mu}(\{k\}) = \begin{pmatrix} \hat{\mu}^1(\{k\}) \\ \vdots \\ \hat{\mu}^C(\{k\}) \end{pmatrix}, \quad \mu(\{k\}) = \begin{pmatrix} \mu^1(\{k\}) \\ \vdots \\ \mu^C(\{k\}) \end{pmatrix}$$

$$\hat{\nu}^d = \begin{pmatrix} \hat{\nu}^d(\{1\}) \\ \vdots \\ \hat{\nu}^d(\{C\}) \end{pmatrix}, \qquad \nu^d = \begin{pmatrix} \nu^d(\{1\}) \\ \vdots \\ \nu^d(\{C\}) \end{pmatrix}.$$

We will bound the probability of the following being true from below:

$$\begin{aligned}
\left|\hat{\pi}^d(\{k\}) - \pi^d(\{k\})\right| &= \left|\left\langle \hat{\mu}(\{k\}), \hat{\nu}^d \right\rangle - \left\langle \mu(\{k\}), \nu^d \right\rangle\right| \\
&= \left|\left\langle \hat{\mu}(\{k\}), \hat{\nu}^d \right\rangle - \left\langle \hat{\mu}(\{k\}), \nu^d \right\rangle + \left\langle \hat{\mu}(\{k\}), \nu^d \right\rangle - \left\langle \mu(\{k\}), \nu^d \right\rangle\right| \\
&\leq \left|\left\langle \hat{\mu}(\{k\}), \hat{\nu}^d - \nu^d \right\rangle\right| + \left|\left\langle \hat{\mu}(\{k\}), \nu^d \right\rangle - \left\langle \mu(\{k\}), \nu^d \right\rangle\right| \qquad (1) \\
&\leq Ct + \left|\left\langle \hat{\mu}(\{k\}) - \mu(\{k\}), \nu^d \right\rangle\right| \\
&\leq Ct + \left\|\hat{\mu}(\{k\}) - \mu(\{k\})\right\|_\infty \left\|\nu^d\right\|_1 \\
&\leq Ct + t = (C+1)t \qquad (2)
\end{aligned}$$

Where Equation (1) holds with probability at least $1 - 2C\exp\left(-2t^2\left|D^d\right|\right)$ and Equation (2) holds with probability at least $1-2C\exp\left(-2t^2\left|D_\ell\right|\right)$, so that the entire derivation holds with probability at least

$$1 - 2C\exp\left(-2t^2\left|D_\ell\right|\right) - 2C\exp\left(-2t^2\left|D^d\right|\right).$$

If we define $s = \min\left\{\min_\ell\left\{\left|D_\ell\right|\right\}, \left|D^d\right|\right\}$, we get that the above holds with probability at least

$$1 - 2C\exp\left(-2t^2\left|D_\ell\right|\right) - 2C\exp\left(-2t^2\left|D^d\right|\right) \geq 1 - 4C\exp\left(-2t^2 s\right).$$

We can solve this to equal $1-\alpha$ and get

$$4C\exp\left(-2t^2 s\right) = \alpha \iff t = \sqrt{\frac{\log(4C/\alpha)}{2s}}.$$

Therefore, $(C+1)t = \sqrt{\frac{(C+1)^2\log(16/\alpha)}{2s}}$ and thus

$$\mathbb{P}\left(\left|\hat{\pi}^d(\{k\}) - \pi^d(\{k\})\right| \geq \sqrt{\frac{(C+1)^2\log\left(4C/\alpha\right)}{2s}}\right) \leq \alpha.$$

Result follows from realising that the value has to be in $[0,1]$. □

In the special case that $C = 2$, we can improve the confidence intervals introduced in Lemma 1;

**Lemma 2** (Confidence Intervals when $C = 2$)**.** *Let $s_d$ and $\alpha$ be as in Lemma 1, and define*

$$c_{d,k}(\alpha) = \sqrt{\frac{2\log\left(6/\alpha\right)}{s_d}}.$$

*When we have only $C = 2$ classes, it holds that*

$$C_{d,k}(\alpha) = \left[\max\left\{\hat{\pi}^d(\{k\}) - c_{d,k}(\alpha), 0\right\}, \min\left\{\hat{\pi}^d(\{k\}) + c_{d,k}(\alpha), 1\right\}\right]$$

*is a $1-\alpha$ confidence interval for $\pi^d(\{k\})$;*

$$\mathbb{P}\left(\pi^d(\{k\}) \in C_{d,k}(\alpha)\right) \geq 1 - \alpha.$$

*Proof.* The main observation is that when $C = 2$ we only need to compute $\hat{\nu}^d(\{k\})$ for $k = 1$ *or* for $k = 2$. This is true because

$$\sum_{k=1}^{2} \left(\hat{\nu}^d(\{k\}) - \nu^d(\{k\})\right) = \sum_{k=1}^{2} \hat{\nu}^d(\{k\}) - \sum_{k=1}^{2} \nu^d(\{k\}) = 0$$
$$\Rightarrow \left(\hat{\nu}^d(\{1\}) - \nu^d(\{1\})\right) = -\left(\left(\hat{\nu}^d(\{2\}) - \nu^d(\{2\})\right)\right).$$

Therefore, we do not need to apply the union bound to $\left|\hat{\nu}^d(\{k\}) - \nu^d(\{k\})\right|$, but rather obtain

$$\begin{aligned}\mathbb{P}\left(\exists k \in \{1,2\} : \left|\hat{\nu}^d(\{k\}) - \nu^d(\{k\})\right| \geq t\right) &= \mathbb{P}\left(\left|\hat{\nu}^d(\{1\}) - \nu^d(\{1\})\right| \geq t\right) \\ &= \mathbb{P}\left(\left|\hat{\nu}^d(\{2\}) - \nu^d(\{2\})\right| \geq t\right) \\ &\leq 2\exp\left(-2t^2\left|D^d\right|\right).\end{aligned}$$

We use the same bounds as in the proof of Lemma 1 for $\left|\hat{\mu}^\ell(\{k\}) - \mu^\ell(\{k\})\right|$. This observation also allows us to improve our bounds on $\left|\hat{\pi}^d(\{k\}) - \pi^d(\{k\})\right|$. To that end, note that Equation (1) from the above proof still holds. However, we can now write

$$\begin{aligned}\left|\left\langle\hat{\mu}(\{k\}), \hat{\nu}^d - \nu^d\right\rangle\right| &= \left|\sum_{l=1}^{2}\left(\hat{\nu}^d(\{l\}) - \nu^d(\{l\})\right)\hat{\mu}^l(\{k\})\right| \\ &= \left|\hat{\nu}^d(\{1\}) - \nu^d(\{1\})\right|\left|\hat{\mu}^1(\{k\}) - \hat{\mu}^2(\{k\})\right| \\ &\leq t\end{aligned}$$

where in the last line we used that $\hat{\mu}^1(\{k\}) - \hat{\mu}^2(\{k\}) \in [-1, 1]$ and $\left|\hat{\nu}^d(\{1\}) - \nu^d(\{1\})\right| \leq t$. Therefore, we have $\left|\hat{\pi}^d(\{k\}) - \pi^d(\{k\})\right| \leq 2t$ with probability at least

$$1 - 2C\exp\left(-2t^2\left|D_\ell\right|\right) - 2\exp\left(-2t^2\left|D^d\right|\right) \geq 1 - 2(C+1)\exp\left(-2t^2 s\right).$$

Following the same approach as in the above proof and setting $C = 2$, we obtain the upper bound on $\left|\hat{\pi}^d(\{k\}) - \pi^d(\{k\})\right|$

$$2t = \sqrt{\frac{4\log(6/\alpha)}{2s}} =: c_{d,k}(\alpha).$$

The rest of the proof is identical to the proof of Lemma 1. □

**Corollary 1** (No uncertainty in $\mu^\ell(\{k\})$, C=2)**.** *Note that if we do not have to estimate $\mu^\ell(\{k\})$, then we can further improve $c_{d,k}(\alpha)$ in Lemma 2 by a factor of 2;*

$$c_{d,k}(\alpha) = \sqrt{\frac{\log(6/\alpha)}{2s}}.$$

*Proof.* This follows again from Equation (1), since if $\hat{\mu}^\ell(\{k\}) = \mu^\ell(\{k\})$,

$$\begin{aligned}\left|\hat{\pi}^d(\{k\}) - \pi^d(\{k\})\right| &\leq \left|\left\langle\hat{\mu}(\{k\}), \hat{\nu}^d - \nu^d\right\rangle\right| + \left|\left\langle\hat{\mu}(\{k\}), \nu^d\right\rangle - \left\langle\mu(\{k\}), \nu^d\right\rangle\right| \\ &= \left|\left\langle\hat{\mu}(\{k\}), \hat{\nu}^d - \nu^d\right\rangle\right|.\end{aligned}$$

Since $C = 2$, we have that $\left|\left\langle\hat{\mu}(\{k\}), \hat{\nu}^d - \nu^d\right\rangle\right| \leq t$, as shown in the proof of Lemma 2. The rest of the proof is identical to that of Lemma 2. □

**Lemma 3** (Bonferroni)**.**

$$\mathbb{P}\left(\pi^d(\{k\}) \in C_{d,k}\left(\frac{\alpha}{m}\right) \ \forall d = 1, ..., m\right) \geq 1 - \alpha$$

*Proof.*

$$\begin{aligned}
\mathbb{P}\left(\pi^d(\{k\}) \in C_{d,k}\left(\frac{\alpha}{m}\right) \ \forall d = 1, ..., m\right) &= \mathbb{P}\left(\bigcap_{d=1}^{m}\left\{\pi^d(\{k\}) \in C_{d,k}\left(\frac{\alpha}{m}\right)\right\}\right) \\
&= 1 - \mathbb{P}\left(\bigcup_{d=1}^{m}\left\{\pi^d(\{k\}) \notin C_{d,k}\left(\frac{\alpha}{m}\right)\right\}\right) \\
&\geq 1 - \sum_{i=1}^{m}\mathbb{P}\left(\pi^d(\{k\}) \notin C_{d,k}\left(\frac{\alpha}{m}\right)\right) \\
&\geq 1 - m\frac{\alpha}{m} = 1 - \alpha.
\end{aligned}$$

□

# B Sample Size Planning

Figure 12 shows some plots of $s_d$ versus $c_{d,k}(\alpha)$ for $C = 2$, using the notation from Appendix A.

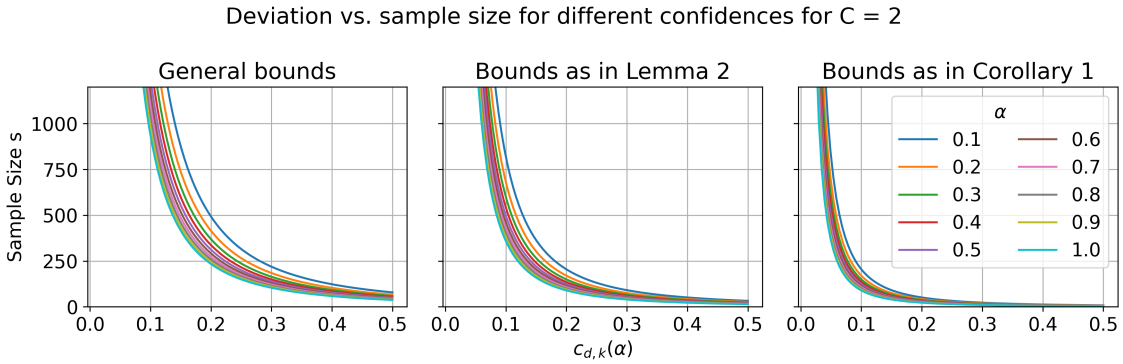


Figure 12: Necessary sample sizes for some confidence intervals using the results of Lemma 1 (*General bounds*), Lemma 2 (*Lemma 2*) and Corollary 1 (*Corollary 1*).

For example, assuming that all of the assumptions in the proofs and for the statistical model hold, if we want that $\left|\hat{\pi}^d(\{k\}) - \pi^d(\{k\})\right|$ is at most 0.2 with probability at least 0.9, then we should look at the curve where $\alpha = 1 - 0.9 = 0.1$, and obtain the following necessary sample sizes:

- Lemma 1 says that we need $s \geq 500$ (roughly),
- Lemma 2 says that we need $s \geq 200$ (roughly),

- Corollary 1 says that we need $s \geq 50$ (roughly).[4]

Recall that $s$ is the minimum of the number of times the least frequently predicted label is predicted, and the size of the dataset for the drug in question.

## B.1 A Word of Caution

We note that the required sample sizes are very different and that it is crucial to verify the assumptions made before drawing any conclusions. In particular,

- for all of the bounds, we have to accept the independence assumption in the population model in Section 3.1,
- comparing the results of Lemma 2 and Corollary 1, we see that a given sample size can give us an $\alpha$ which is twice as good, if we accept that we can estimate a quantity *perfectly*, which we may not be able to do in practice, and finally
- comparing the *General bounds* of Lemma 1 to the others, this improvement is only possible when we have exactly two classes.

However, when the computed bounds show that the uncertainty in $\left|\hat{\pi}^d(\{k\}) - \pi^d(\{k\})\right|$ is large, i.e. that the model cannot distinguish the effectiveness of drugs with high confidence, then there is reason to believe that we need more data. We recommend a healthy scepticism towards optimistic bounds, but do believe that there is value to reporting the uncertainty about these estimates.

[4] Of course, using the results from Appendix A, we could compute the sample sizes exactly. This example just indicates how to interpret Figure 12.